MARK D. FOWLER, Bar No. 124235
mark.fowler@dlapiper.com
CLAYTON THOMPSON (Admitted Pro Hac Vice)
clayton.thompson@dlapiper.com
DAVID ALBERTI, Bar No. 220625
david.alberti@dlapiper.com
CHRISTINE K. CORBETT, Bar No. 209128
christine.corbett@dlapiper.com
YAKOV M. ZOLOTOREV, Bar No. 224260
yakov.zolotorev@dlapiper.com
CARRIE L. WILLIAMSON, Bar No. 230873
carrie.williamson@dlapiper.com

DLA PIPER US LLP
2000 University Avenue
East Palo Alto, CA 94303-2214
Tel: 650.833.2000
Fax: 650.833.2001

Attorneys for Defendant and Counterclaim Plaintiff
SUN MICROSYSTEMS, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

NETWORK APPLIANCE, INC.,

Plaintiff-Counterclaim Defendant,

v.

SUN MICROSYSTEMS, INC.,

Defendant-Counterclaim Plaintiff.

CASE NO. C-07-06053-EDL

SUN MICROSYSTEMS, INC.'S OPENING CLAIM CONSTRUCTION BRIEF

Claim Construction Hearing:
Date: August 27, 2008
Time: 9:00 a.m.
Judge: Hon. Elizabeth D. Laporte

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

TABLE OF AUTHORITIES

Page

CASES

TABLE OF AUTHORITIES
(continued)

Page

STATUTES

I. INTRODUCTION

This brief addresses the construction of the claim terms identified by Sun for construction in NetApp's United States Patent Nos. 5,819,292 (the "'292 patent"), 6,892,211 (the "'211 patent") and 7,200,715 (the "'715 patent"). Pursuant to the Court's June 17, 2008, Order Regarding Claim Construction, the parties currently are briefing the claim terms identified as 1 through 14 in the parties' June 16, 2008, Joint Report to the Court. Pursuant to the briefing protocol established at the June 4, 2008, Case Management Conference, this brief addresses the seven claim terms identified by Sun for construction, while NetApp's opening brief will address the seven terms identified by NetApp for construction. Each party will address the seven claim terms identified by the other party in its July 21, 2008, responsive claim construction brief.

II. APPLICABLE RULES OF CLAIM CONSTRUCTION

The usual rules of claim construction, reviewed at length by the Federal Circuit in *Phillips v. AWH Corporation*, 415 F.3d 1303 (Fed. Cir. 2005), apply. In *Phillips*, the Federal Circuit confirmed the primacy of the intrinsic evidence in claim construction, focusing in particular on the language of the claims and the specification. *Phillips*, 415 F.3d at 1312-1317. Sun's proposed constructions are anchored in the intrinsic evidence.

A few of Sun's proposed constructions involve the specific application of particular rules of claim construction. Those rules are addressed below in connection with the constructions to which they specifically apply.

III. U.S. PATENT NO. 5,819,292

A. Technology Background.[1]

1. Structure Of The File System.

The '292 patent describes features of a file system based on the Write Anywhere File-system Layout (WAFL). '292 patent, col. 5:46-49. The system employs disks formatted to use four kilobyte (4KB) blocks that cannot be fragmented. *See, e.g.,* '292 patent, col. 5:49-51. In

[1] A copy of the '292 patent is attached as Exhibit A to the Declaration of Carrie L. Williamson ("Williamson Decl."). A detailed overview of the underlying technology of the '292 patent is provided at paragraphs 35-60 of the Declaration of Dr. Scott Brandt ("Brandt Decl.").

WAFL's block-based format system, the storage blocks on the disk (the 4KB blocks with no fragments), also referred to as "disk blocks," are configured to be the same size as the 4KB "data blocks" used by the file system. *Id.*, col. 6:18-53, Figs. 4B-4D. Disk drives provide the storage space for the file system and maintain the structure and content of the file system even when power to the system is interrupted. *Id.*, col. 1:18-19, col. 20:25-28.

The file system also requires an in-memory "buffer" for holding changes to the file system prior to storing them on the disk. '292 patent, col. 6:56-60, Fig. 8. This in-memory storage must be configured to handle the equivalent of the 4KB data blocks that are stored on the disk. *Id.*, col. 6:58-60.

The 4KB data blocks are organized and described by 4KB "inode" blocks. '292 patent, col. 5:51-53. Each inode "points to" 16 4KB blocks. *Id.*, col. 5:60-64, col. 6:18-20, Fig. 4B. The file system is scalable to store large data files by using multiple levels of blocks. As illustrated below by Figure 4B of the patent, if the data file is small, such as a file size of less than 64 KB, the inode points directly to the 4KB data blocks. *Id.*

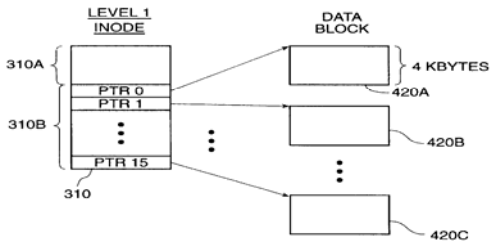

FIG. 4B

If the data file is larger than 64 KB, then scalability is achieved by having each of the inode's 16 block pointers point to an "indirect" block. Depending upon the size of the file, each indirect block then either points to a data block or to another indirect block. *Id.*, col. 6:23-53, Figs. 4C-4D. This is illustrated below by Figure 4D of the '292 patent, which shows three levels of inode blocks:

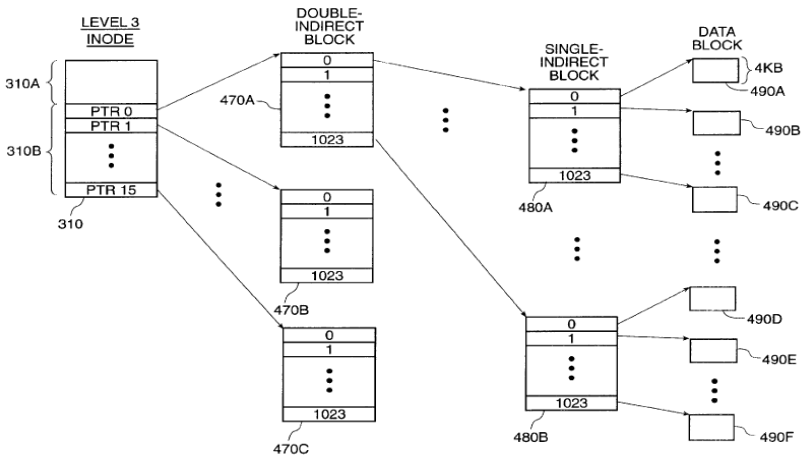

FIG. 4D

In Figure 4D, the 16 pointers of the "Level 3 Inode" point to a maximum of 16 "Double Indirect Blocks." Each of the 16 Double Indirect Blocks has 1024 block numbers which in turn point to 1024 "Single Indirect Blocks" (so the 16 Double Indirect Blocks collectively point to a maximum of 16,384 Single Indirect Blocks). Each Single Indirect Block also has 1024 block numbers, each of which points to a 4KB data block (so the Single Indirect Blocks collectively point to a maximum of 16,777,217 4KB data blocks). In this way, up to 64 GB of data blocks can be addressed. *Id.*, col. 6:35-52, Fig. 4D.

The inodes are stored in an "inode file" in the non-volatile storage. '292 patent, col. 9:22-28. The inode file "contains an inode 1210A-1210F for each file in the file system …." *Id.*, col. 9:30-32, Fig. 12. The inode file (containing all of the inodes that locate all of the files of the file system) may be written on any disk location. *Id.*, col. 9:27-29. That freedom requires a "root inode" to point to the inode file, and that the root inode be "kept in a fixed location on disk referred to as the file system information (fsinfo) block." *Id.*, col. 9:32-35. Otherwise, without being able to find the root inode at a fixed location, WAFL would be incapable of locating the file system following a system crash. Brandt Decl., ¶ 58; '292 patent, col. 10:57-66, col. 14:1-10.

2. Moving To A New Consistent State.

When all of the 4KB data blocks are written to disk and are identified by the file system, the file system is said to be "consistent." '292 patent, col. 11:60-12:1. Changes to the file system occur during its use. *Id.*, col. 4:8-9, col. 11:62-12:8. One feature of the '292 patent is that the

current file system on the disks remains consistent even while modifications are being made to the data and written to disks. *Id.*, col. 11:65-66. Until the file system commences preparation of a new consistency point, the modified blocks (described as "dirty" blocks) are accumulated and organized in the in-memory buffer blocks. '292 patent, col. 7:17-27, col. 11:29-59.

"To implement [the next] consistency point[], WAFL always writes new [or modified] data to unallocated blocks on disk [so that it] never overwrites existing data." '292 patent, col. 12:2-4, col. 4:13-15. To establish a new consistency point, all of the modified blocks other than the fsinfo block are first written to disk. *Id.*, col. 14:24-27, col. 12:9-20. The '292 patent requires that the fsinfo block be written last because it contains the root inode that roots the file system. *Id.*, col. 10:57-66, col. 12:9-11, col. 14:24-27 ("otherwise the entire file system . . . could be lost"). "A new consistency point occurs when the fsinfo block is updated by writing a new root inode …" *Id.*, col. 4:16-18, col. 12:2-6, 18-20. By triggering the new consistency point by writing the fsinfo, the file system progresses to a new consistency point in one transaction ("atomically") and thereby retains tight control over the file system. '292 patent, col. 12:25-26, col. 14:26-28, col. 4:8-19, col. 20:23-27 ("The file system of the present invention is completely consistent as of the last time the fsinfo blocks . . . were written. Therefore, if power is interrupted to the system, upon restart the file system . . . comes up in a consistent state.")

3. Snapshots.

The file system described in the '292 patent can retain a copy of a past consistent state in a read-only form called a snapshot. '292 patent, col. 4:20-21, col. 17:64-18:3. A snapshot is similar to a past consistency point. *Id.*, col. 20:21-22, col. 64-65. The presence of a snapshot ensures the "old" data blocks associated with the snapshot are not overwritten. *Id.*, col. 4:33-36, col. 19:29-33. Thus, snapshots can be used to backup file systems while continuing to allow access to the file system or to restore data that has been lost in the active file system. Brandt Decl., ¶ 59.

The '292 patent emphasizes the value of being able to maintain multiple snapshots. '292 patent, col. 3:66-4:3, col 20:28-30. "WAFL supports up to 20 different snapshots that are numbered 1 through 20." *Id.*, col. 18:8-9. The ability to record multiple snapshots is enabled by

providing numerous bits for each block in a block map table. *Id.*, col. 20:15-17. "The key data structures for snapshots are the blkmap entries where each entry has multiple bits for a snapshot. This enables a plurality of snapshots to be created." *Id.* By setting a bit (equal to 1) in a block's entry in the block map, the file system ensures the block will not be overwritten and that it will remain part of a snapshot. *Id.*, col. 18:26-30.

B. Representative Claims.

Claims 4 and 8 are the two independent claims of the '292 patent asserted by NetApp against Sun. These claims are reproduced below with the disputed claim terms in bold.

> 4. A method for maintaining a file system comprising blocks of data stored in blocks of a **non-volatile storage means** at successive consistency points comprising the steps of:
>
> storing a first **file system information structure** for a first consistency point in said **non-volatile storage means**, said first **file system information structure** comprising data describing a layout of said file system at said first consistency point of said file system;
>
> writing blocks of data of said file system that have been modified from said first consistency point as of the commencement of a second consistency point to free blocks of said **non-volatile storage means**;
>
> storing in said **non-volatile storage means** a second **file system information structure** for said second consistency point, said second **file system information structure** comprising data describing a layout said file system at said second consistency point of said file system.

'292 patent at 25:11-29 (emphasis added).

> 8. A method for creating a plurality of read-only copies of a file system stored in blocks on a **non-volatile storage means**, said file system comprising meta-data identifying blocks of said **non-volatile storage means** used by said file system, comprising the steps of:
>
> storing **meta-data for successive states of said file system** in said **non-volatile storage means**;
>
> making a copy of said meta-data at each of a plurality of said states of said file system;
>
> for each of said copies of said meta-data at a respective **state of said file system**, marking said blocks of said **non-volatile storage means** identified in said meta-data as comprising a respective read-only copy of said file system.

Id., col. 26:1-15 (emphasis added).

C. Disputed Claim Terms To Be Construed.

1. "nonvolatile storage means"

Sun's proposed construction	NetApp's proposed construction
This limitation is a means-plus-function limitation governed by 35 U.S.C. §112, ¶ 6. Function: storing blocks of data of a file system so that the data is not lost in the absence of power Structure: one or more disks with a block-based format (i.e., 4KB blocks that have no fragments), where the disk storage blocks are the same size as the data blocks of the file system	A storage device that can retain information in the absence of power.

The critical issue here is whether the term "nonvolatile storage means" is a means-plus-function limitation under 35 U.S.C. §112, ¶6. Because the term includes the word "means," it is presumed to be governed by section 112, ¶6. Further, the claims do not cite sufficient structure to rebut the presumption. Therefore, Sun's construction should be adopted.

a. This Term Is A Means-Plus-Function Limitation.

Section 112, ¶6 permits applicants to recite elements of a claim "as a means or step for performing a specified function without the recital of structure material or acts in support thereof" in the claim itself. 35 U.S.C. §112, ¶6. When one chooses to use such "means" language in a claim, "[t]he price that must be paid for use of that convenience is limitation of the claim to the means specified in the written description and equivalents thereof." *Texas Digital Sys., Inc. v. Telegenix, Inc.*, 308 F.3d 1193, 1208 (Fed. Cir. 2002) (quoting *O.I. Corp. v. Tekmar Co.*, 115 F.3d 1576, 1583 (Fed. Cir. 1997)); 35 U.S.C. ¶ 112, ¶ 6 ("such claim shall be construed to cover the corresponding structure, material or acts described in the specification and equivalents thereof").

It is well-settled the use of the word "means" in a claim limitation raises a rebuttable presumption the limitation is a means-plus-function limitation under section 112, ¶6. *Altiris, Inc. v. Symantec Corp.*, 318 F.3d 1363, 1375 (Fed. Cir. 2003), *citing Sage Prods., Inc. v. Devon*

Indus., Inc., 126 F.3d 1420, 1427 (Fed. Cir. 1997). This presumption may be rebutted by showing the claim language itself recites sufficient ***structure*** to perform the claimed function in its entirety. *Id.* The recitation of some structure in a means-plus-function limitation does not preclude the applicability of section 112, ¶6. *Laitram Corp. v. Rexnord, Inc.*, 939 F.2d 1533, 1536 (Fed. Cir. 1991). Rather, the claim language must recite a specific physical structure that performs the function. *Altiris,* 318 F.3d at 1376. Here, no such specific physical structure exists in either claim 4 or 8.

The expressly recited functions of the "nonvolatile storage means" in claims 4 and 8 is to store data blocks of a file system (claims 4 and 8), to store first and second "file system information structures" (claim 4), to store "read-only copies of a file system" (claim 8) and to store "metadata for successive states of said file system" (claim 8). '292 patent, col. 25:11-29, col. 26:1-15. The claims do not recite any structure, much less specific structure, to perform any (much less all) of these recited functions. Accordingly, this claim term must be construed as a means-plus-function limitation. *Altiris*, 318 F.3d at 1376.

If NetApp argues the term "nonvolatile storage means" itself describes the requisite specific physical structure, NetApp is incorrect. The word "nonvolatile" is an adjective and describes a result – not losing data when power is lost. Brandt Decl., ¶ 64. The word "nonvolatile" does not identify any specific physical structure. *Id.* Similarly, while the word "storage" is a noun, it does not equate with any specific physical structure. *Id*.

Nor does the combined phrase "nonvolatile storage" define, to one of ordinary skill in the art, any particular physical structure or class of devices for performing the recited functions. Brandt Decl., ¶ 65. Instead, myriad unique and completely unrelated structures can be used to retain information in a way that does not require power. *Id*. For example, paper, tape, punch cards, hard disks, battery-backed RAM devices, ROM devices and countless other unrelated devices can be used to store information in a manner that can retain information in the absence of electrical power. *Id*. Thus, the term "nonvolatile storage means" does not connote a particular physical structure or class of physical structures.

The Federal Circuit has held "means" limitations that arguably include some structural

language, but do not recite a specific structure for performing the entire function, do not overcome the presumption that section 112, ¶6 applies. For example, in *Altiris*, the Federal Circuit concluded the term "a means of booting said digital computer" was a means-plus-function limitation even though the claim recited "said means for booting including a first set of commands . . . and a second set of commands." The Court concluded that "[a]lthough commands represented structure (in the form of software), it is not sufficient structure to perform the entirety of the function." *Altiris*, 318 F.3d at 1376. Similarly, in *Unidynamics Corp. v. Automatic Prods. Int'l Ltd.*, 157 F.3d 1311 (Fed. Cir. 1998), which addressed the claim term "spring means tending to keep the door closed," the Federal Circuit concluded that although the limitation included the structural term "spring," that did not provide sufficient structure to overcome the presumption that section 112, ¶6 applied. *Unidynamics*, 157 F.3d at 1319-20. Likewise, in *Overhead Door Corp. v. Chamberlain Group, Inc.*, 194 F.3d 1261 (Fed. Cir. 1999), the Federal Circuit found the terms "memory selection second switch means" and "first switch means" invoked section 112, ¶6 despite the use of the structural term "switch" in the claims. *Overhead Door*, 194 F.3d at 1272-75.

In the present case, the term "non volatile storage means" provides less structure – more specifically, no structure – than was the case in *Altiris*, *Unidynamics* or *Overhead Door*. Accordingly, the Court should conclude, as the Federal Circuit did in those cases, that the presumption has not been overcome and, therefore, that the term "nonvolatile storage means" is subject to section 112, ¶6.

b. The Function Of The Nonvolatile Storage Means.

The construction of a means-plus-function limitation involves two steps. First, the Court must identify the claimed function. *See Omega Eng'g, Inc. v. Raytek Corp.*, 334 F.3d 1314, 1321 (Fed. Cir. 2003). Second, the Court must identify the corresponding structure in the specification that performs the function. *Id.*

As established above, the expressly recited functions of the "nonvolatile storage means" in claims 4 and 8 are to store data blocks of a file system (claims 4 and 8), to store first and second "file information structures" (claim 4), to store "read-only copies of a file system" (claim 8) and

to store "metadata for successive states of said file system" (claim 8). '292 patent, col. 25:11-29, col. 26:1-15. The read-only copies of the file system will inherently include data blocks. Brandt Decl., ¶ 63. Moreover, because the storage is described as "nonvolatile," the stored data is not lost in the absence of power. Brandt Decl., ¶ 62, Ex. 3 (Webster's Dictionary of Computer Terms, 8th Ed. at 376). Consequently, the stated function for the term "nonvolatile storage means" in ***both*** claims 4 and 8 includes, ***at a minimum***, "storing blocks of data for a file system so that the data is not lost in the absence of power."

c. The Corresponding Structure Of The Nonvolatile Storage Means.

A structure disclosed in the specification is "corresponding" if the specification clearly links or associates that structure to the function recited in the claim. *Omega Eng'g*, 334 F.3d at 1321. Here, the specification discloses ***only one*** corresponding structure performing the recited function of storing data blocks of a file system so the data is not lost in the absence of power – and that disclosure is quite clear.

The specification defines the file system of the invention as a Write Anywhere File-system Layout (WAFL). '292 patent, col. 5:48-50 ("The present invention uses a Write Anywhere File-system Layout (WAFL)"). And, in fact, the specification repeatedly, ***and exclusively***, describes the invention in the context of the structures and operation of a WAFL system. *See, e.g.*, '292 patent, col. 5:45-6:52 (describing "On-Disk WAFL Inodes"), col. 6:53-8:56 (describing "Incore WAFL Inodes"), col. 8:57-9:17 (describing WAFL directories), col. 9:18-11-27 (describing WAFL meta-data files), col. 11:28-58 (describing WAFL inodes having "dirty" blocks), col. 11:62-17:63 (discussing consistency points in a WAFL system), col. 17:64-24:6 (discussing snapshots in a WAFL system).

Fundamental to WAFL is its use of a "disk format system [that] is block based (i.e., 4KB blocks that have no fragments)." '292 patent, col. 5:48-53; Brandt Decl., ¶ 67. In the WAFL block-based disk format disclosed in the specification, the storage blocks on the disk (the 4KB blocks having no fragments) are configured to be the same size as the 4KB data blocks that make up the files of the file system. *Id.*, col. 6:18-53, Figs. 4B-4D; Brandt Decl., ¶ 67. Significantly,

this use of disks with a block based format (using 4KB blocks having no fragments), with storage blocks the same size as the data blocks used in the file system, is ***the only structure disclosed*** for storing blocks of data of a file system in a location where the data is not lost in the absence of power. Brandt Decl., ¶ 67. Therefore, the Court should find that the required "corresponding structure" is "one or more disks with a block-based format (*i.e.*, 4KB blocks that have no fragments), where the disk storage blocks are the same size as the data blocks of the file system."

2. "meta-data for successive states of said file system"

Sun's proposed construction	NetApp's proposed construction
a block map file for recording snapshots of the file system	information that describes "successive states of a file system" (as construed herein)

This claim term appears in claim 8. Claim 8 pertains to creating "read-only copies" of successive consistent states of the file system. '292 patent, col. 26:1-15; Brandt Decl., ¶ 69. The patent specification defines such read-only copies of the file system as "snapshots." '292 patent, col. 17:65-18:1 ("A snapshot is a read-only copy of an entire file system at a given instant when the snapshot is created."), col. 4:21-22 ("The present invention also creates snapshots, which are virtual read-only copies of the file system.").

A necessary step in creating a snapshot is ensuring blocks of any snapshot are not lost by being overwritten. *Id.*, col. 4:33-36. This necessary step is accomplished by setting multiple bits ***within a block map file***. *Id.*, col. 4:33-40. Claim 8 recites this necessary step as storing the "meta-data for successive states of said file system," the claim term now at issue. *Id.*, col. 26:6.

a. The Claim Language Requires Sun's Construction.

To create a "plurality" of snapshots, claim 8 recites multiple "steps" for "identifying [the] blocks" in the file system to be recorded by the snapshot. '292 patent, col. 26:1-5. Specifically, claim 8 requires that "meta-data" identify each block, that the meta-data be stored and copied to create each snapshot, and that the snapshot mark the blocks identified by the meta-data. *Id.*, col. 26:3-12. This claim language ***necessarily*** describes what one skilled in the art understands is a "block map file" where each block is identified for each snapshot. Brandt Decl., ¶ 70.

Dependent claims 9 and 10 (not asserted by NetApp) narrow and further define (for those

claims) the type of block map file to be used for recording snapshots. Specifically, claims 9 and 10 require that the block map file must record "multiple usage bits per block." '292 patent, col. 26:20-25.

b. The Specification and Prosecution History Require Sun's Construction.

It is well-established "the specification 'is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.'" *Phillips*, 415 F.3d at 1315 (quoting *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)). This certainly is true here, as the "Summary of the Invention" section of the specification clearly states:

> The ***present invention*** prevents new data written to the active file system from overwriting "old" data that is part of snapshot(s). It is ***necessary*** that old data not be overwritten as long as it is part of a snapshot. ***This is accomplished by using a multi-bit free-block map***. Most prior art file systems use a free block map having a single bit per block to indicate whether or not a block is allocated. ***The present invention uses a block map*** having 32-bit entries.

'292 patent, col. 4:33-40 (emphasis added).

These unambiguous statements – which refer to the "invention" and not merely to an embodiment of the invention –make clear a block map is a necessary feature of the invention. The rest of the specification is in accord. Indeed, the ***only*** disclosure in the specification for using "metadata for" recording "successive states of a file system" – the claim term at issue – is by using block maps.

The specification states the file system "uses files to store meta-data that describes the layout of the file system." '292 patent, col. 5:53-55. The allocation of blocks within the file system is identified within the block map (blkmap) file. *Id.*, col. 5: 57-58, col. 9:54-55. Indeed, "[t]he ***key*** data structures for snapshots are the ***blkmap*** entries where each entry has multiple bits for a snapshot." *Id.*, col. 20:15-17 (emphasis added).

The block map file is described in great detail in the specification as assigning a 32-bit entry to each block. *Id.*, col. 9:51-53. The block map file contains meta-data associated with snapshots, as shown below in the Figure 21E from the specification:

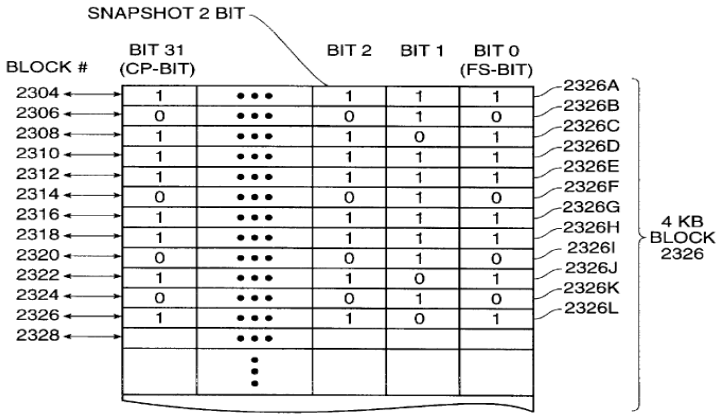

FIG. 21E

As illustrated above, each block is identified in the block map file and is given a block number, here between the range 2304 through 2328. *Id.*, col. 9:51-53, col. 16:22-23. For each block, the first 21 bits (bits 0-20) identify whether a block is part of the current state of the file system or of any snapshots. *Id.*, col. 9:59-65. The value stored at "BIT 0" indicates whether the block is part of the active file system. *Id.*, col. 10:7-9. "BIT 1" is the first snapshot bit, "BIT 2" is the second snapshot bit, and so on through bit 20. *Id.*, col. 16:29-20, col. 23:2-5. The block map file indicates a block is part of the active file system or a snapshot by storing a "1" in the bit position. *Id.*, col. 10:7-15. If a block is not part of the active file system or part of a snapshot, a "0" is entered into the block map file. *Id.*, col. 9:66 – 10:1.

The specification describes ***only*** this one method for tracking blocks as snapshots. *Id.*, col. 4:39-43; Brandt Decl., ¶ 74. In such a case, where the specification describes only one embodiment, especially where, as here, a necessary feature is described as being part of the "invention," claim 8 is properly read as requiring the use of a block map file. *Honeywell Int'l., Inc. v. ITT Indust., Inc.*, 452 F.3d 1312, 1318-19 (Fed. Cir. 2006) (limiting the disputed claim term to the single embodiment disclosed in the specification where the embodiment was referred to as "this invention" or the "the present invention"); *Verizon Serv. Corp. v. Vonage Holdings, Corp.*, 503 F.3d 1295, 1308 (Fed. Cir. 2007) ("When a patent thus describes the features of the 'present invention' as a whole, this description limits the scope of the invention."); *Decisioning.com, Inc. v. Federated Dept. Stores, Inc.*, 527 F.3d 1300, 1311 (Fed. Cir. 2008). To

borrow the Federal Circuit's conclusion in *Honeywell* after it reviewed the specification in that case, "[t]he public is entitled to take the patentee at his word and the word was that the invention" uses a block map file. *Honeywell*, 452 F.3d at 1318.

NetApp's proposed construction errs in two respects. First, it omits the block map file required by the intrinsic evidence. Second, by substituting the generic term "information" for the claim term "meta-data," NetApp reads the "meta-data" requirement out of the claim. In this regard, the specification identifies which files are meta-data, and also states that not all "information" is meta-data. '292 patent, col. 9:21-22, col. 10:60-62; Brandt Decl., ¶ 75. Obviating an entire claim limitation, as NetApp does by replacing "meta-data" with information, is forbidden. *Innova/Pure Water, Inc. v. Safari Water Filtration Syst., Inc.*, 381 F.3d 1111, 1119 (Fed. Cir. 2004) (holding "all claim terms are presumed to have meaning in a claim"); *Merck & Co., Inc. v. Teva Pharm. USA, Inc.*, 395 F.3d 1364, 1372 (Fed. Cir. 2005) (holding "[a] claim construction that gives meaning to all terms of the claim is preferred over one that does not do so" and reversing a construction that rendered a claim term "superfluous"); *Mangosoft, Inc. v. Oracle Corp.*, 525 F.3d 1327, 1330-31 (Fed. Cir. 2008) (citing *Merck* in rejecting a "proposed construction ascribes no meaning to the [disputed claim term] not already implicit in the rest of the claim").

c. The Related '352 Patent Confirms The Correctness Of Sun's Construction.

After allowance of the '292 patent, NetApp filed multiple continuations and continuations-in-part, which eventually resulted in the issuance of U.S. Patent No. 7,174,352 (the "'352 patent"). '352 patent, col. 1:6-13. A copy of the '352 patent, which NetApp is asserting against Sun in this case, is attached as Ex. D to the Williamson Declaration.

The '352 patent expressly incorporates by reference several patent applications, including the application that resulted in the '292 patent, and summarizes their disclosure, referring to them as the "WAFL Disclosures." '352 patent, col. 2:40-44, 48-50. As such, the '352 patent is a reliable guide as to the meaning of the claim terms of the '292 patent.

The specification of the '352 patent states that the earlier NetApp patents (including the

'292 patent) describe that "when changes to the file system [] are committed to the mass storage [], the block map is altered to show those storage blocks [] that are part of the committed file system []." '352 patent, col. 4:31-35. The block map is modified to reflect storage blocks forming a "(consistent) file system [] at some point in time." *Id.*, col. 5:22-25. A snapshot is a "cop[y] of the file system [] that [is] read-only." *Id.*, col. 5:31-34. Snapshots "can be created based on the block map at any time by copying bits from the … blocks [that] are part of the [current] file system [] into the corresponding bits [] for the snapshot[]." *Id.*, col. 5:22-28. The "snapshot is stored as a file system object, such as a blockmap." *Id.*, col. 4:57-61.

Thus, the applicants' own statements in the '352 patent concerning the subject of claim 8 of the '292 patent confirms that: (1) a block map is the meta-data used to describe successive states of the file system; and (2) the read-only copies of these states are described as snapshots. These admissions confirm the compelling intrinsic evidence in the '292 patent supporting Sun's proposed construction.

3. "file system information structure"

Sun's proposed construction	NetApp's proposed construction
data structure that contains the root inode of a file system in a fixed location on disk	a data structure containing information about the layout of a file system

This claim term appears in asserted claim 4, as well as in non-asserted claims 2, 5 and 7. The specification repeatedly and unequivocally equates this claim term – "file system information structure" – with the "fsinfo" block. The specification also defines these two equivalent terms the same way Sun does: a data structure at a fixed location on disk that contains the root inode of the file system. '292 patent, col. 9:33-36, col. 10:57-60. Because the patent is quite intentional about its use of the "file system information structure," Sun's construction must be adopted.

a. The Specification Expressly Equates The "File System Information Structure" With The "FSINFO" Block.

The "file system information structure" is unequivocally defined by the specification. Brandt Decl., ¶¶ 78-79. The specification repeatedly equates the file system information structure with the "fsinfo" block. This is done in part by placing the parenthetical definition "(fsinfo)" next

to or within the phrase "file system information structure." For example, the specification twice refers to the "file system information (fsinfo) structure" and twice cites the "file system information (fsinfo) block." '292 patent, col. 9:33-36, col. 10:57-65, col. 13:63-64. *See Irdeto Access, Inc. v. Echostar Satellite Corp.*, 383 F.3d 1295, 1301 (Fed. Cir. 2004) (construing "group" as a subscriber subset based in part on the specification's "most telling" reference to "subscriber set (group)").

The specification also equates the terms "file system information structure" and "fsinfo block" (as well as the term "fsinfo structure") by using them interchangeably. *See, e.g.*, col. 11:3-4, col. 12:27-37. For example, the specification states "FIG. 15 is a diagram illustrating a file system information (fsinfo) structure 1510." *Id.*, col. 10:57-58. Figure 15 labels structure 1510 "fsinfo block." *Id.*, Fig. 15. This is seen throughout the specification – the "fsinfo block" serves the same function in Figures 16, 18A-18C, 20B, 21A-21D, 21F, 22 and 23A, and in the corresponding text of the specification relating to those figures. *Id.*, col. 11:6-27, col. 18:50-19:2. In short, the '292 patent consistently defines the "file system information structure" as, and uses it synonymously with, the "fsinfo block."

b. The Specification Dictates The File System Information Structure Be Located At A Fixed Location And Include The Root Inode.

The specification is equally clear in stating the necessary requirements of the file system information structure. As discussed above at pages 3 and 4 of the "Technology Background," a fundamental aspect of the alleged invention of the '292 patent is the structure, function and location of the "root inode." In this regard, the specification is crystal clear in requiring the "file system information structure" be kept in a fixed location and include the root inode: "The root inode is kept in a fixed location on disk referred to as the file system information (fsinfo) block described below." '292 patent, col. 9:33-35. The specification again confirms this requirement: "FIG. 15 is a diagram illustrating a file system information (fsinfo) structure 1510. The root inode 1510B of a file system is kept in a fixed location on disk so that it can be located during booting of the file system . . . The root inode 1510B is an inode referencing the inode file 1210. It is part of the file system information (fsinfo) structure 1510 . . ." *Id.*, col. 10:57-64; *see also*

Figs. 20A-20C, 21A-21D, 21F (each showing the "fsinfo block" as including the "root inode").

The phrase "file system information structure" is not commonly used within the industry, and does not by itself have an established meaning within the industry. Brandt Decl., ¶ 82. However, based upon the language of claim 4 and the unequivocal meaning ascribed to this claim term by the specification, a person of ordinary skill in the art recognizes the "file system information structure" must include the root inode and must be located at the fixed location. *Id.*, ¶ 84.

For example, the specification teaches the use of the file system information structure to "progress[] from one self-consistent state to another self-consistent state." '292 patent, col. 4:11-12. "A new consistency point occurs when the fsinfo block is updated by writing a new root inode …." *Id.*, col. 4:16-18. "The … file system 1670 reflects an old consistency point up until the root inode . . . is written. Immediately after the root inode . . . is written to disk, the file system 1670 reflects a new consistency point." *Id.*, col. 12:16-20. As such, a person of ordinary skill in the art knows the file system information structure cannot operate to change consistency points absent the presence of the root inode in that structure. Brandt Decl., ¶ 83.

The fixed, known location of the fsinfo block is necessary to claim 4. Brandt Decl., ¶ 84. Recovery of the file system after a power interruption is only possible if the root inode can be located by the file system. '292 patent, col. 14:22-29, col. 13:66-14:3; Brandt Decl., ¶ 84. This requirement, together with the specification's unequivocal teaching that the file system information structure must be located at a fixed location (col. 9:34-35, col. 10:57-65, col. 11:3-5), requires that the structure both include the root inode and be at a fixed location. '292 patent, col. 14:22-20; Brandt Decl., ¶ 84.

c. NetApp's Construction Is Inconsistent With The Surrounding Claim Language And The Specification.

NetApp's construction once again attempts to strip any meaning from the limitation. NetApp's construction treats each part of the claim limitation – file system – information – and structure – as though they have no relationship to each other, and as if the specific teaching of the "file system information structure" in the specification is irrelevant.

NetApp's proposed construction – a data structure containing information about the layout of a file system – is not really a construction at all. It provides no guidance as to the meaning of the claim term other than merely repeating the word "information" from the claim and then stating the information concerns "the layout of a file system." However, claim 4 itself states the "file system information structure compris[es] data describing a layout of said file system." '292 patent, col. 16-19. Indeed, if NetApp's construction were accepted, the claim language as construed would read "said [data structure containing information about the layout of a file system] comprising data describing a layout of said file system …." As a matter of law, the proper construction of a claim term cannot be derived simply by pulling the surrounding claim language into the claim term, so as to create a tautology within the claim. *See AllVoice Computing PLC v. Nuance Communications, Inc.*, 504 F.3d 1236, 1247-48 (Fed. Cir. 2007) (refusing to define a limitation to encompass a second limitation in the same claim); *Mangosoft*, 525 F.3d at 1330-31 (rejecting a proposed construction that would have rendered the disputed claim term "superfluous" and "ascribe[d] no meaning to the [disputed claim term] not already implicit in the rest of the claim"). Because NetApp's proposed construction violates this rule, it must be rejected.

NetApp's construction also must be rejected because it appears to encompass any kind of inode disclosed in the specification – not just root inodes. Brandt Decl., ¶ 85 However, there is no support in the specification for such a construction *Id.*, ¶ 85.

IV. U.S. PATENT NO. 6,892,211

A. Technology Background.[2]

The '211 patent is a continuation of the '292 patent and is directed to a method for keeping a file system in a consistent state. '211 patent, Abstract, col. 1:11-13. The '211 and '292 patents share the same specification. While claim 4 of the '292 patent (discussed above) claims a file system that progresses from one consistent state to another, the '211 patent presents a more

[2] A copy of the '211 patent is attached as Exhibit B to the Williamson Declaration. Sun's expert concerning the '211 patent, Dr. Brandt, provides a detailed overview of the underlying technology in paragraphs 24-29 and 87-132 of the Brandt Declaration.

detailed view of the mechanics of that progression.

In the '211 patent file system, files are stored as blocks on disk. '211 patent, col. 5:50-54. Each file may be broken up into multiple blocks of 4KB each, but each file is uniquely described by its corresponding inode. *Id*., col. 5:50-54, 62-65. An inode is a data structure containing meta-data – information about the file such as ownership, file size, access time, etc. *Id*., col. 5:53-54, col. 6:4-7.

An inode contains 16 pointers which, depending upon the size of the data file, point either directly to data blocks or to a set of indirect blocks (which in turn ultimately point to data blocks). *Id*., col. 6:19-50, Figs. 4B-4D. This is explained above at pages 2 and 3 of this brief (in the Technology Background section concerning the '292 patent). As previously explained, adding additional levels of indirect blocks allows an inode to serve as the root of progressively larger data files. This file system structure – using inodes to root a multi-level "tree" of blocks, with data blocks as the "leaves" on the outer branches of the tree – is common to many file systems and is well known in the art. Brandt Decl. ¶ 91.

The '211 patent file system, WAFL (Write Anywhere File-system Layout), writes file data blocks and inodes to unallocated blocks on disk, wherever such blocks are available. '211 patent, col. 4:18-21, 5:50-54. One consequence of such "write-anywhere" data placement is that the file system needs a mechanism for tracking the location of files on storage, otherwise the file system would "lose" files. Brandt Decl., ¶ 92-97. This problem is well known in the art and is solved by the '211 patent file system (and prior art file systems) by having a data structure located at a fixed, known location on disk that serves as the starting point for any file access operations. *Id*., ¶ 97. As discussed above in connection with the '292 patent, this data structure is the root inode. '211 patent, col. 4:15-18, col. 9:32-35, col. 10:58-65.

The root inode contains pointers that directly point to a special meta-data file, the inode file, that contains the inodes of all of the other files in the file system. '211 patent, col. 9:25-33 ("A first meta-data file is the 'inode file' that contains inodes describing, all other files in the file system . . . The inode file 1210 is pointed to by an inode referred to as the 'root inode.'"); col. 10:62-63. The figure in the '211 patent that best illustrates this is Figure 17B:

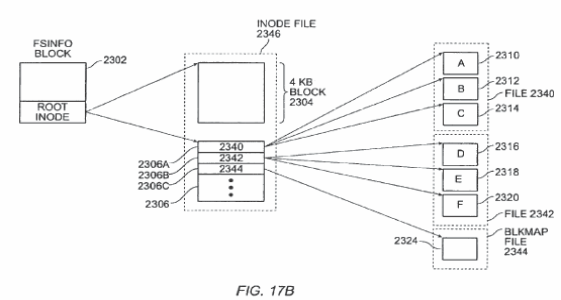

FIG. 17B

In Fig. 17B, the root inode is located in the fsinfo block 2302.[3] The root inode points to blocks of the inode file 2346, which contain file inodes that point to files 2340, 2342 and BLKMAP (block map) file 2344, a special meta-data file discussed above in connection with the '292 patent (and discussed further below). '211 patent, col. 15:55-16:3. Thus, the root inode points directly to blocks of the inode file and indirectly to all of the other file system blocks. *Id.*, col. 11:20-25, Fig. 16; Brandt Decl., ¶ 98-99. This feature is claimed in the following language of asserted claim 1 of the '211 patent:

> maintaining an on-disk root inode on said storage system, said on-disk root inode pointing directly and indirectly to a first set of blocks on said storage system that store a first consistent state of said file system.

'211 patent, col. 23:64-68.

In order to make modifications to data stored on the disk while at the same time keeping the disk in a "consistent state" (discussed above in connection with the '292 patent), the '211 patent file system "caches" (temporarily places) a copy of the inodes and their data blocks in "incore" (in memory) buffers. *Id.*, col. 6:55-7:17. Like other inodes, the fsinfo block storing the on-disk root inode is copied to an incore buffer, creating an "incore root inode." *Id.*, col. 15:30-31, col. 16:2-3, Figs. 17A-17L (illustrating the buffer holding the root inode). Each incore inode contains the information of the corresponding on-disk inode, including its block pointers. *Id.*, col. 6:55-7:17, col. 7:6-17, Fig. 10. The incore (in memory) buffers are thus used by the '211 patent file system (and prior art file systems) as a "holding pen" for data undergoing changes before such data is written to permanent storage, such as a disk. Brandt Decl., ¶ 100-101.

[3] This block is the "file system information structure" claimed in the '292 patent, the proper construction of which is discussed above.

B. Representative Claims.

The '211 patent includes three independent claims – claims 1, 9 and 17 – each of which is asserted against Sun. Claims 1, 9 and 17 have substantially similar limitations. Claim 9 is reproduced below with the disputed claim terms to be construed identified in bold.

> 9. A device comprising:
>
> a processor;
>
> a memory; and
>
> a storage system including one or more hard disks;
>
> wherein said memory and said storage system store a file system; and
>
> wherein said memory also stores information including instructions executable by said processor to maintain said file system, the instructions including steps of (a) maintaining an on-disk **root inode** on said storage system, said on-disk **root inode** pointing directly and indirectly to a first set of blocks on said storage system that store a first **consistent state** of said file system, and (b) maintaining an incore **root inode** in said memory, said incore **root inode pointing directly and indirectly to buffers in said memory and a second set of blocks on said storage system**, said buffers and said second set of blocks storing data and metadata for a second **consistent state** of said file system, said second set of blocks including at least some blocks in said first set of blocks, with changes between said first **consistent state** and said second **consistent state** being stored in said buffers and in ones of said second set of blocks not pointed to by said on-disk inode.

'211 patent, col. 24:39-62.

C. Disputed Claim Terms To Be Construed.

1. "pointing directly and indirectly to buffers in said memory and a second set of blocks on said storage system"

Sun's proposed construction	NetApp's proposed construction
pointing directly and indirectly to buffers in said memory and pointing directly and indirectly to a second set of blocks on said storage system	No construction necessary, plain and ordinary meaning. To the extent the Court deems a construction necessary, the term means pointing directly to blocks and/or buffers, and/or indirectly to blocks and/or buffers.

This term appears in independent claims 1, 9 and 17. Amazingly, the parties' dispute is the meaning of the word "and" in the claim term. Sun's construction gives "and" its regular

conjunctive meaning, while NetApp reads "and" as "and/or." This difference is significant because a normal reading of the claim language (Sun's construction) requires each part of the claim limitation be satisfied, whereas NetApp's contorted reading of the claim would permit any one of at least eight conditions to satisfy the claim.

a. Sun's Construction Is Mandated By The Plain Claim Language.

The three independent claims each plainly require that the "incore root inode point[] directly ***and*** indirectly to buffers in said memory ***and*** a second set of blocks on said storage system." '211 patent, col. 24:2-4, 53-55, col. 25:34-36 (emphasis added). This language mandates that the incore root inode ***both*** point directly to some buffers ***and*** point indirectly to other buffers. The plain claim language ***also*** requires the incore root inode to point directly to some blocks ***and*** indirectly to other blocks. There is no other reasonable reading of the claims.

Sun's plain-meaning construction is directly supported by the preceding language in each claim requiring the on-disk root inode to "point[] directly ***and*** indirectly to a first set of blocks." *Id.*, col. 23:65-66, col. 24:49-50, col. 25:31-32 (emphasis added). Since, as described above, the incore root inode is a copy of the on-disk root inode and, therefore, has the same property of pointing directly ***and*** indirectly to blocks, there is no doubt the claim language intends exactly what it actually says – that the incore root inode points directly and indirectly to buffers and points directly and indirectly to blocks.

NetApp's proposed construction, by replacing two "ands" with three "and/ors", renders the claim term unrecognizable. If NetApp's construction were adopted, any one of at least the following conditions would satisfy the claim language:

- pointing directly to buffers;
- pointing directly to blocks;
- pointing indirectly to buffers;
- pointing indirectly to blocks;
- pointing directly to blocks and buffers;
- pointing indirectly to blocks and buffers;
- pointing directly to blocks and indirectly to buffers; or
- pointing directly to buffers and indirectly to blocks.

Thus, under NetApp's litigation-driven construction, infringement might be found so long as the

incore root inode points to something, a block or a buffer, in any way. This plainly is not what the claim language says, and it is not how one of ordinary skill in the art understands the claim. Brandt Decl., ¶ 123.

b. Sun's Construction Is Supported By The Specification.

When read in the context of the surrounding claim language, it is clear the buffers and blocks to which the incore root inode points serve a specific purpose – to "stor[e] data and metadata for a second consistent state of said file system." '211 patent, col. 24:4-6, 55-57, col. 25:37-38. With that purpose in mind, it is easy to confirm the specification teaches that the incore root inode must point directly ***and*** indirectly to buffers and blocks. Brandt Decl., ¶ 105.

In order to keep data in a consistent state, as required by the claims, the '211 patent file system has to maintain special bookkeeping files, known as meta-data files. *Id*., ¶ 105-106. The '211 patent specification teaches that the file system maintains at least three such files: the inode file; the block map file; and the inode map file. '211 patent, col. 9:19-24. Taken together, these meta-data files represent an irreducible minimum of meta-data required by the claimed invention. Brandt Decl., ¶ 107-110.

The inode file is described above and it is the only file to which the root inode points directly: "[t]he inode file 1210 is pointed to by an inode referred to as the 'root inode'." '211 patent, col. 9:32-33. The inode file contains inodes of all the other files in the file system, including the block map and inode map files. *Id*., col. 9:25-32, col. 11:6-27.

The block map file keeps track of which blocks are allocated in the current file system, which blocks are used by any of the snapshots, and which blocks are free for the file system to write to. *Id*., col. 9:50-65. This meta-data file is critical to keeping the file system in a consistent state; the file system would not be operational without it because it would have no idea as to the usage of its blocks. Brandt Decl., ¶ 108. In fact, the Summary of the Invention section of the specification touts the block map as purportedly distinguishing the '211 patent from the prior art. '211 patent, col. 4:38-48.

The third minimally necessary meta-data file is the inode map file. '211 patent, col. 10:20-49. This file is used by the file system to keep track of which inodes in the inode file are

unallocated. *Id.*

The specification teaches the root inode points to the inode file ***directly*** and to the other two meta-data files (as well as to all regular data files) ***indirectly***: "Because the root inode 1610B and 1612B of the fsinfo blocks 1610 and 1612 describes the inode file 1620, that in turn describes the rest of the files 1630-1660 in the file system including all meta-data files 1630-1640, the root inode 1610B and 1612B is viewed as the root of a tree of blocks." *Id.*, col. 11:20-27. Thus, in order for the root inode to serve its expressly claimed function of rooting "buffers and said second set of blocks storing data and metadata for a second consistent state of said file system," the root inode ***must*** point ***directly*** to the inode file ***and*** point ***indirectly*** to the meta-data files (block map and inode map) as well as the regular data files. Brandt Decl., ¶ 111.

When the on-disk root inode is copied as an incore root inode, it points directly to buffers containing blocks of the inode file and indirectly to buffers containing changed data files, as shown in Figure 17L, reproduced below. Figure 17L shows that data block 2320 of file 2342 containing data F has been modified, so that an incore buffer 2322 containing changed data F′ has been allocated:

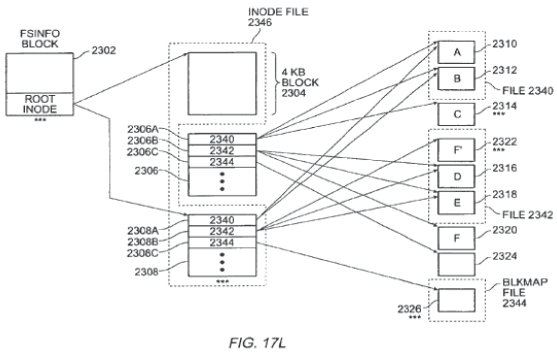

FIG. 17L

Allocating a new buffer for data F′ has, in turn, caused the allocation of a buffer 2308 holding a modified inode for file 2342, with the modified inode pointing to buffer 2322 as well as to unmodified blocks 2316 and 2318. '211 patent, col. 16:4-17:57. Thus, the incore root inode stored in the fsinfo block 2302: (1) points directly to the inode file buffer 2308 holding inodes for modified files 2340, 2342 and the block map file 2344; and (2) simultaneously points indirectly to buffers holding modified data for these files. *Id.* Once these buffers are "flushed" (sent) to disk, the incore root inode directly and indirectly points to a new set of blocks comprising the

changed data and inodes. '211 patent, col. 17:35-57. At the same time, the incore root inode still points directly to unchanged blocks of the inode file (*e.g.* block 2304 in Fig. 17L) and indirectly to the corresponding unchanged data blocks (not shown in Fig. 17L). Thus, the specification unambiguously teaches to one of skill in the art that the incore root inode points directly and indirectly to buffers in said memory and points directly and indirectly to a second set of blocks on said storage system, as required by Sun's construction. Brandt Decl., ¶ 112-120.

This transition from one consistent state to another, before the new root inode is written to disk to generate a new consistency point ('211 patent, col. 11:61-12:13), is recited in claim 1 as follows:

> maintaining an incore root inode in said memory, said incore root inode ***pointing directly and indirectly to buffers in said memory and a second set of blocks*** on said storage system, said buffers and said second set of blocks storing data and metadata for a second consistent state of said file system, said second set of blocks including at least some blocks in said first set of blocks, with changes between said first consistent state and said second consistent state being stored in said buffers and in ones of said second set of blocks not pointed to by said on-disk inode.

'211 patent, col. 24:1-11 (emphasis added).

NetApp may point to locations in the specification, such as col. 7:55-57, where it discusses *regular* inodes and suggests that such inodes may point to "indirect and/or direct blocks."[4] This is inapposite as regular inodes do not play the unique role the specification ascribes to the root inode – rooting a consistent file system. Brandt Decl., ¶ 122. The specification describes no embodiment where the incore root inode points directly to blocks ***without also*** pointing indirectly to blocks and pointing directly to buffers ***without also*** pointing indirectly to buffers, as would be permitted by NetApp's strained construction. In fact, as discussed above, the only embodiment disclosed in the '211 patent teaches a structure that requires the incore root inode to point directly ***and*** indirectly to blocks and to buffers in order to maintain a consistent file system state. Brandt Decl., ¶ 120-121. Accordingly, the Court should

[4] Indeed, this is the only place in the specification where the term "and/or" is used with respect to how an inode may reference data blocks.

adopt Sun's proposed construction.

2. "root inode"

Sun's proposed construction	NetApp's proposed construction
the index node data structure stored in a fixed location that roots a set of self-consistent blocks on the storage system that comprise the file system	an inode that points directly and/or indirectly to all the blocks in a consistent state of a "file system" (as construed herein).

The term "root inode" appears in independent claims 1, 9 and 17. The fundamental difference between the parties' respective constructions is whether the root inode resides in a fixed location. Because the patent unmistakably requires the root inode to reside in a fixed location, the Court should adopt Sun's construction.

a. Sun's Construction Is Supported By The Intrinsic Evidence As Understood By One Of Ordinary Skill.

The '211 patent specification teaches (and the claims require) that the root inode roots a set of blocks that comprise the file system in a consistent state. '211 patent, col. 11:20-27, 11:65-67, col. 23:64-67, col. 24:48-51, col. 25:30-33. In fact, there is no disagreement between the parties on this point. However, NetApp's construction fails to recognize that a fundamental aspect of the root inode is that it must be stored in a fixed location.

As understood by one of skill in the art, the term "root" as applied to a file system means a fixed starting point that allows the file system to find any of the file system blocks without prior knowledge of their location. Brandt Decl., ¶ 125-126. A file system lacking knowledge of the location and usage of file system blocks is common – an example being when a file system restarts after a crash has wiped out or corrupted the incore file system structures storing the up-to-date file system information. *Id*. Without such information, the file system is forced to go back to square one – a fixed, known location where its root is located. *Id*. In the '211 patent, the need for a fixed, known root is satisfied by the claimed root inode.

The specification explicitly teaches "[t]he root inode 1510B of a file system is kept in a fixed location on disk so that it can be located during booting of the file system." '211 patent, col. 10:59-61. The fixed location is where the fsinfo block is located: "[t]he root inode is kept in

a fixed location on disk referred to as the file system information (fsinfo) block described below." *Id.*, col. 9:33-35. Two copies of the fsinfo structure containing the root inode are stored in fixed locations on disk for reliability. *Id.*, col. 11:3-5. To transition from one consistent state to another, the root inode is sent from the incore cache to these two fixed locations on disk. *Id.*, col. 12:1-23.

As explained in the Technology Background section above, the need for a fixed root is critical in the "write-anywhere" file system taught by the '211 patent. If the incore root inode and required meta-data files (*e.g.*, the inode file and the block map) are lost due to a crash, the file system would lose track of the location of all of its files absent a fixed root – the claimed on-disk root inode. Brandt Decl., ¶ 127-129; '211 patent, col. 14:22-25. This is because blocks of the inode file, which contain all of the file inodes, are written anywhere on disk, with their location constantly changing as data in the file system is modified. '211 patent, col. 9:25-35. The specification thus teaches that the on-disk root inode – the only inode that points to (locates) the inode file – must be at a fixed location. *Id.*; Brandt Decl., ¶ 128-130. This fundamental aspect of what makes a root inode the root of the file system is absent from NetApp's proposed construction.

The specific language of Sun's construction – that the root inode "roots a set of self-consistent blocks on the storage system that comprise the file system" – is supported by the specification. The Summary of the Invention teaches that "[t]he file system progresses from one self-consistent state to another self-consistent state. The set of self-consistent blocks on disk that is rooted by the root inode is referred to as a consistency point (CP)." '211 patent, col. 4:15-18. Thus, the specification describes a file system in a consistent state ("a set of self-consistent blocks") rooted by a root inode. *See also* '211 patent, col. 11:61-67. This teaching is reflected in Sun's proposed construction.

b. NetApp's Construction Incorrectly Ties Its Construction Of "Root Inode" With NetApp's Other Incorrect Claim Constructions.

NetApp's construction incorrectly ties the construction of "root inode" with NetApp's incorrect construction of the term "directly and indirectly" (discussed above) by requiring the root

inode to "point[] directly ***and/or*** indirectly" to other blocks. This clearly is incorrect as the claim language itself states the "on-disk root inode point[s] directly ***and*** indirectly to a first set of blocks" storing a consistent file system state. '211 patent, col. 23:65-66, col. 24:48-50, col. 25:31-31 (emphasis added).

NetApp's proposed construction has a third flaw. Specifically, by using the phrase "'file system' (as construed herein)" in its construction, NetApp apparently requires adoption of its construction of "file system" in its construction of "root inode." There are two problems with this. First, Sun contends NetApp's construction of "file system" is incorrect. Second, the term "file system" is not being construed by the Court at this time. Therefore, if the Court were to adopt NetApp's proposed construction of "root inode" (which it should not, for the reasons discussed above), the Court's construction would reference a construction of "file system" that does not yet exist.

3. "consistent state"/"state of a file system"[5]

Sun's proposed construction	NetApp's proposed construction
a set of storage blocks for that file system that includes all blocks required for the data and structure of the file system	a set of blocks on disk, rooted by a "root inode" (as construed herein), that includes all the blocks required for the data and file structure of a "file system" (as construed herein)

The parties' respective constructions of these claim terms are substantively identical except that NetApp: (1) adds the phrase "rooted by a 'root inode' (as construed herein)"; and (2) adds the phrase "(as construed herein)" after the words "file system." Significantly, Sun's proposed construction is identical to NetApp's express definition of the term "consistent" in NetApp's '352 patent, a continuation in part of the '292 patent. The '352 patent shares most of

[5] The term "consistent state" appears in the '211 patent, while the term "state of a file system" appears in the '292 patent. The parties agree the construction of these two claim terms should be the same. Because the '211 patent is a continuation of the '292 patent, the two patents share the same specification. Therefore, the language in the '211 patent specification that supports Sun's construction of "consistent state" also supports Sun's construction of "state of a file system" in the '292 patent.

its inventors with the '292 and '211 patents and its disclosure concerns NetApp's WAFL file system – the same file system as disclosed in the '292 and '211 patents. Williamson Decl., Ex. D, '352 patent. The '352 patent includes the following express definition of "consistent": "As used herein, the term 'consistent,' referring to a file system (or to storage blocks in a file system), means a set of storage blocks for that file system that includes all blocks required for the data and file structure of that file system." '352 patent, col. 4:24-27. This definition is identical to Sun's proposed construction and should be adopted here by the Court.

NetApp's current, litigation driven construction should be rejected for two reasons. First, NetApp's proposed construction expressly incorporates by reference the construction of "file system." As discussed above in connection with NetApp's construction of "root inode," the Court is not construing "file system" at this time. Thus, construing "consistent state" to refer to a construction of "file system" that does not yet exist would be incorrect.

Second, NetApp's inclusion of the phrase "rooted by a 'root inode' (as construed herein)" in its proposed construction of "consistent state" creates a confusing tautology because "root inode" already is a separate claim limitation. '211 patent, col. 23:64-24:6, 48-49, 52-53, col. 25:30-31, 33-34. Specifically, if one inserts NetApp's construction of "root inode" into its construction of "consistent state" (as NetApp's proposed construction essentially does), and if one then inserts the resulting construction of "consistent state" into the place of "consistent state" in the relevant claim language (*e.g.* '211 patent, claim 1 at 23:64-67), the following incoherent statement results:

> maintaining an on-disk inode that points directly and/or indirectly to all the blocks in a consistent state of a 'file system' (as construed herein) on said storage system, said on-disk inode that points directly and/or indirectly to all the blocks in a consistent state of a 'file system' (as construed herein) pointing directly and indirectly to a first set of blocks on said storage system that store a first set of blocks on disk, rooted by an inode that points directly and/or indirectly to all the blocks in a consistent state of a 'file system' (as construed herein), that includes all blocks required for the data and file structure of a 'file system' (as construed herein) of said file system.

This impenetrable language provides no meaningful guidance to the Court, the jury or the parties as to the meaning of this claim term. Because identifying the meaning of claim language is the purpose of claim construction, NetApp's construction – which creates uncertainty rather

than eliminates it – should be rejected.

V. U.S. Patent No. 7,200,715.

A. Technology Background.[6]

The '715 patent is directed to a system and method for controlling data storage over arrays of disk drives. '715 patent, col. 1:7-9. As described by the patent, multiple disk drives can be connected together to form a "redundant array of inexpensive/independent disks," otherwise known as a "RAID" system. *Id.*, col. 1:27-32. RAID systems can provide greater storage capacity than a traditional single disk drive system because additional disk drives can be added to the array. *Id.*, col. 1:32-34, 41-43.

RAID systems also have high data transfer rates. *Id.*, col. 1:32-36. This is because, in a RAID system, data files are broken down into data blocks, with each data block being written to a different disk in the array. Because the data blocks are written to different disks, they can be accessed at the same time, in parallel, thereby allowing simultaneous access to all data blocks in a data file. *Id.*, col. 1:27-30, 32-41. RAID systems also offer greater reliability. This is because if a single disk in the array fails, it can be replaced without shutting down the entire system. *Id.*, col. 1:32-36, 43-45.

As explained by the '715 patent, data blocks are written across the array of disk drives in "stripes." *Id.*, col. 1:37-38. The patent defines a "stripe" as including "one storage block on each disk drive in an array of drives in the system." *Id.*, col. 1:37-39. Each block in a particular stripe is located at the same offset (disk block number) on every disk in the array. *Id.*, col. 1:50-53. Figure 1A from the '715 patent illustrates the RAID system described in the patent:

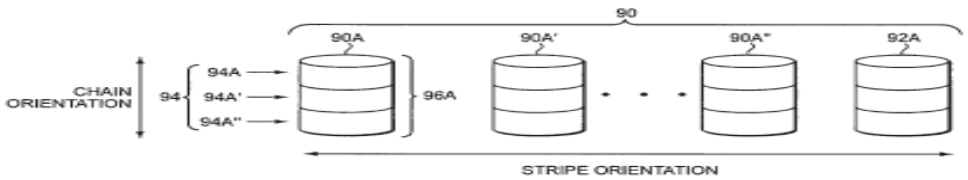

FIG. 1A

In the RAID system of the '715 patent, data is written over an array of disks 90A, 90A' and 90A''. Each disk includes a number of storage blocks used to store data written to the disk. Data is written across the disk array as a series of stripes 94A, 94A' and 94A''. *Id.*, col. 4:65-5:21. The RAID system also includes a parity disk 92A. The parity value for each of the stripes is written on the parity disk. *Id.*, col. 5:1-2, 8-9.

Parity information can be used to reconstruct a data file from a failed disk. *Id.*, col. 59-61. Using Figure 1A as an example, if the data file in question consists of three data blocks of information, the data file can be written across stripe 94A, with one data block being written to one storage block on disk 90A, one data block being written to a storage block on disk 90A', and one data block being written to a storage block on disk 90A''. Parity is then calculated for stripe 94A and stored in a corresponding storage block of parity disk 92A. *See, e.g., id.* at col. 6:35-58. As a result, if disk 90A' fails, the data file can be retrieved by accessing the storage blocks on data disks 90A and 90A'' and on parity disk 92A. The parity information on disk 92A is used with the data on disks 90A and 90A'' to recreate the data on failed disk 90A'.

The alleged invention of the '715 patent purports to improve prior RAID systems in which the file system layer sent *individual stripe* writes. *Id.*, col. 9:28-29, col. 11-14-21. In the alleged invention, the file system layer generates "block layout information," also referred to in the patent as an "association," for "associating" data blocks with storage blocks over *multiple stripes. Id.*, col. 5:48-67, col.. 9:13-31, col. 11:22-31. As explained in the specification, the association maps each data block to one storage block in the association. *Id.*, col. 9:13-19. In response to the "association," the data blocks are written to the associated storage blocks in a single write request. *Id.*, col. 6:1-8, col. 9: 23-31, col. 12:30-34.

B. Representative Claims.

NetApp asserts three independent claims of the '715 patent, and a number of dependent claims, against Sun. Each of the asserted independent claims – claims 21, 39 and 52 – includes

---

(footnote continued from previous page)

[6] A copy of the '715 patent is attached as Exhibit C to the Williamson Declaration.

the disputed claim phrase to be construed by the Court. Representative claims 21 and 39 are reproduced below with the disputed phrase identified in bold.

> 21. A method for controlling storage of data, comprising:
>
> receiving one or more write requests associated with data blocks;
>
> receiving topological information associated with storage blocks configured in a plurality of parallel stripes of a storage system;
>
> **associating the data blocks with one or more storage blocks across the plurality of stripes as an association**; and
>
> writing the data blocks, in response to the association, to the one or more storage devices in a single write request.

'715 patent, col. 21:41-52 (emphasis added).

> 39. A storage system, comprising:
>
> a file system, the file system to receive one of more write requests associated with data blocks;
>
> a storage device manager, the storage device manager to generate topological information of storage blocks configured in a plurality of parallel stripes of one or more storage devices, and to send the topological information to the file system; and
>
> an association generated in the file system, **the association to associate the data blocks with one or more storage blocks across the plurality of stripes** of the one or more storage devices, the association to be sent to the storage device manager, the storage device manager to write the data blocks, in response to the association, to the one or more storage blocks as a single write request.

Id., col. 22:24-39 (emphasis added).

C. Disputed Claim Terms To Be Construed.

1. "associating the data blocks with one or more storage blocks across the plurality of stripes as an association" (claims 21 and 52) and "the association to associate the data blocks with one or more storage blocks across the plurality of stripes" (claim 39)

Because these two claim phrases are substantively identical (only the verb tense of "to associate" and the location of word "association" in the two phrases differ), they are addressed as one by Sun for purposes of claim construction. As noted immediately below, NetApp offers separate constructions for the two phrases, although the two NetApp constructions are

substantively identical in the context of the other claim language.

Sun's proposed construction	NetApp's proposed constructions
Sun contends this phrase is indefinite under 35 U.S.C. § 112, ¶2 and cannot reasonably be construed because it fails to particularly point out and distinctly claim the subject matter which applicants regard as their invention. However, if the Court decides the phrase is not indefinite, the term should be construed to mean: "associating each data block with a respective one of the storage blocks across the plurality of stripes as an association."	(Claims 21 and 52) No construction necessary, plain and ordinary meaning. To the extent the Court deems a construction necessary, the term means "creating a data structure that relates data blocks to locations on more than one stripe." (Claim 39) No construction necessary, plain and ordinary meaning. To the extent the Court deems a construction necessary, the term means "the data structure relating data blocks to locations on more than one stripe."

a. The Claim Terms Are Indefinite Under 35 U.S.C. §112, 2nd Paragraph.

The claims of a patent must "particularly point[] out and distinctly claim[] the subject matter which the applicant regards as his invention." 35 U.S.C. § 112, 2nd paragraph. "It is the applicants' burden to precisely define the invention, not the PTO's," and section 112, ¶ 2 "puts the burden of precise claim drafting squarely on the applicant." *In re Morris*, 127 F.3d 1048, 1056 (Fed. Cir. 1997) ("The problem in this case is that the appellants failed to make their intended meaning explicitly clear.").

A claim is indefinite under section 112, ¶ 2, "[w]here it would be apparent to one of skill in the art, based on the specification, that the invention set forth in the claim is not what the patentee regarded as his invention." *Allen Eng'g Corp. v. Bartell Indus., Inc.*, 299 F.3d 1336, 1349 (Fed. Cir. 2002). Thus, if after comparing the claims and the specification, the Court concludes the claims do not recite the invention described in the specification, the claim is invalid as a matter of law under section 112, ¶ 2. *Id.*

A claim also is indefinite under section 112, ¶ 2 when it is "inherently inconsistent" or when it "its legal scope is not clear enough that a person of ordinary skill in the art could determine whether a particular [product] infringes or not." *Geneva Pharms., Inc. v.*

GlaxoSmithKline PLC, 349 F.3d 1373, 1384 (Fed. Cir. 2003). Thus, a claim is invalid when "[t]he claim is 'insolubly ambiguous' and not 'amenable to construction.'" *Novo Indust., L.P. v. Micro Molds Corp.*, 350 F.3d 1348, 1358 (Fed. Cir. 2003) (quoting *Exxon Research & Eng'g. v. United States*, 265 F.3d 1371, 1375 (Fed. Cir. 2001).

Here, the claim phrase "associating the data blocks with ***one or more*** storage blocks across the plurality of stripes as an association" is indefinite because it is inconsistent with the teaching of the invention in the specification, and because it cannot apprise a person of ordinary skill in the art what products infringe the claims. Specifically, while the above-quoted language of claims 21, 39 and 52 requires "one or more storage blocks across the plurality of stripes," it is impossible under this invention for any ***"one"*** storage block to be "across a plurality of stripes."

As established above, the specification expressly defines a stripe: "a stripe includes one storage block on each disk drive in an array of disk drives in the system." '715 patent, col.1:37-39. Thus, as defined by the specification, while "stripes" are written across the disks in an array, each "storage block" (1) is a component of a single stripe, (2) is located on a single disk, and (3) is not written across multiple disks and does not traverse a plurality of stripes. *Id.* This relationship is clearly depicted in Figure 1B and explained in the corresponding text in the specification. *Id.*, col. 6:35-58; *see also* Fig. 1A.

Figure 1B (reproduced below) depicts 5 disks in the array, disks 1 through 4 and parity disk P. Six stripes cross the five-disk array. Each of the 24 boxes depicted in Figure 1B under disks 1-4 is a separate data storage block. *Id.*, col. 6:35-58. No one storage block is located in more than one stripe. This is true in every depiction of the invention in the specification. *See, e.g., id.* at Fig. 6 and col. 13:14-17 (four stripes, each consisting of four storage blocks, each storage block in a stripe located on a different disk); Fig. 8 and col. 13:34-43 (six stripes, each consisting of four storage blocks, each storage block in a stripe located on a different disk).[7]

---

[7] In his recent deposition, one of the named inventors on the '715 patent, Douglas Doucette, acknowledged a given storage block is not part of two different stripes at the same time. Williamson Decl., ¶ 12, Ex. K, p. 133.

NetApp appears to recognize claims 21, 39 and 52 recite the impossible, as NetApp's proposed claim constructions expunge the unfeasible "one or more" language from the claims. Specifically, NetApp defines the existing claim language – "one or more storage blocks across a plurality of stripes" – to mean "to locations on more than one stripe." In doing so, NetApp addresses the inconsistency between the specification and the claim language, and the impossibility of what is claimed, by ***rewriting the claim*** – entirely eliminating the "one or more" claim limitation and changing "data blocks" to mere "locations."

Because claims 21, 39 and 52 contradict the specification's explicit teaching of what constitutes a "stripe" and what constitutes a "storage block," and the relationship between those two elements, and because claims 21, 39 and 52 recite the impossible in the context of the invention as described in the specification, those claims (1) fail to recite what the applicants regarded as their invention and (2) are insoluably ambiguous and, therefore, lack the required clarity to enable one of ordinary skill in the art to determine whether a particular product infringes. As such, claims 21, 39 and 52 are invalid under section 112, ¶ 2. *Allen Eng'g Corp.*, 299 F.3d at 1349; *Geneva Pharms.*, 349 F.3d at 1384; *Novo Indust.*, 350 F.3d at 1358.

NetApp cannot cure the indefiniteness of the claims by writing two claim limitations – "one or more" and "storage blocks" – out of the claims. In this regard, it is well-established the Court cannot rewrite the claims to preserve their validity. *Allen Eng'g Corp*, 299 F.3d at 1349 ("It is not our function to rewrite claims to preserve their validity.") Accordingly, rather than rewrite the substance of claims 21, 52, and 39, as NetApp proposes to do to save them, the Court should find those claims, and each of the asserted dependent claims, indefinite and invalid.[8]

b. If The Court Determines The Claims Are Not Indefinite, Sun's Proposed Claim Construction Should Be Adopted.

Two critical distinctions exist between the parties' respective constructions. First, Sun's proposed construction recognizes, while NetApp's construction does not, the required one-to-one

[8] NetApp asserts the following dependent claims of the '715 patent: 22, 24-25, 33-34 (dependent on claim 21); 41-45, 49-50 (dependent on claim 39); 52 (dependent on claim 51).

correspondence between each data block and an associated storage block. Second, the claims and Sun's proposed construction each require "storage blocks." NetApp's proposed construction, on the other hand, eliminates this express requirement in favor of the broader term "locations."

(i) The Specification And Prosecution History Require A One-To-One Correspondence Between Each Data Block And An Associated Storage Block.

The specification repeatedly explains that data blocks are associated with storage blocks on a one-to-one basis, *i.e.*, each data block is associated with a respective one of the storage blocks. The '715 patent Abstract teaches that "[e]ach data block of the data to be written is associated with a respective one of the storage blocks . . ." '715 patent, Abstract. The specification states the block layout information "associates each data block of the buffered write requests 41 with a storage block in a group of storage blocks." '715 patent, col. 9:16-19; *see also* col. 13:37-39 ("The group includes six stripes in the array 20, which provide 19 free blocks for one-to-one association with the 19 data blocks of the buffered write requests 41."); col. 13:20-22 ("The data blocks and the association are transmitted to, and processed by, the disk array manager 13 so that each data block is stored at its associated storage block in the group."); col. 13:2-5 ("unallocated storage blocks are selected in the array 20A for storage of a corresponding number of data blocks associated with buffered write requests."). This clear teaching of the specification is present in Sun's construction, while it is absent from NetApp's construction.

Moreover, NetApp's representations to the PTO during the prosecution of the patent compel the adoption of Sun's construction. During prosecution, the Examiner rejected all of the proposed claims of the patent, including the claims now being asserted by NetApp, as being anticipated by prior art. In response, NetApp repeatedly argued to the Examiner that the claims of the '715 patent were distinguishable over the prior art because the claims require associating each data block with a respective one of the storage blocks. This construction, which enabled NetApp to obtain the '715 patent, is embodied in Sun's proposed construction.

On May 20, 2004, the Examiner rejected all of the claims of the application as being anticipated by U.S. Patent No. 6,148,368 to DeKoning ("DeKoning"). Williamson Decl., ¶ 6, Ex. E, pp. 1-5. In its August 20, 2004, response to the rejection, NetApp twice urged the Examiner to

allow the claims because "the absence from the DeKoning patent of Applicant's '***associating each data block with a respective one of the storage blocks, for transmitting the association to a storage device manager for processing of the single write transaction***.'" Williamson Decl., ¶ 7, Ex. F, pp. 17-18 (emphasis in original). NetApp further argued:

> Applicant's claimed invention is directed toward the mapping of data blocks with the storage blocks to which they will be written. Applicant claims buffering a plurality of write requests and combining them into a single write transaction. In addition to this, however, **Applicant goes one step further by associating each data block of the single write request with a storage block of the storage system** before transmitting the buffered write request to a storage device manager. In this way, the storage device manager is not required to map each data block to a storage block of the storage system, as it would have conventionally been required to do. …

Id. (emphasis added).

On November 19, 2004, the Examiner again rejected the proposed claims in light of DeKoning. Williamson Decl., ¶ 8, Ex. G. NetApp responded on February 22, 2005, again arguing "that DeKoning does not show Applicant's claimed novel, '***mapping each data block with a respective one of the storage blocks across a plurality of stripes, for transmitting the mapping to a storage device manager for processing of the single write transaction***.'" Williamson Decl., ¶ 9, Ex. H, p. 20 (emphasis in original). NetApp again also argued:

> Applicant's claimed invention is directed toward the mapping of data blocks with the storage blocks to which they will be written. Applicant claims buffering a plurality of write requests and combining them into a single write transaction. In addition to this, however, **Applicant goes one step further by mapping each data block of the single write request with a storage block** *across a plurality of stripes* of the storage system before transmitting the buffered write request to a storage device manager.

Id. (bold emphasis added; italics in original).

After a subsequent Office Action, NetApp again stressed the importance of this aspect of the invention:

> Applicant respectfully urges that all art cited during prosecution of this Application is completely silent regarding ***"associating each data block with a respective one of the storage blocks"*** as claimed. See also Applicant's Specification at Page 19, lines 5-7 and Page 23, lines 21-26.

Williamson Decl., ¶ 10, Ex. I, August 17, 2005, Amendment, p. 15 (emphasis in original).

The Examiner ultimately allowed the claims, including the claims that issued as claims

21, 39 and 52 (*i.e.*, application claims 51, 69 and 82). Significantly, in allowing the claims, the Examiner ultimately accepted NetApp's assertion that the cited prior art does not disclose "associating each [of the] data blocks to be stored with a respective one of the storage blocks across the plurality of stripes for a single write operation." Williamson Decl. ¶ 11, Ex. J, October 28, 2005, Notice of Allowability, p. 2.

This prosecution history conclusively demonstrates the one-to-one correspondence of each data block to a respective storage block was a critical factor in NetApp distinguishing its alleged invention over the cited prior art. Because NetApp repeatedly interpreted the claims to require associating "each data block with a respective one of the storage blocks across the plurality of stripes" in order to overcome prior art rejections, NetApp cannot now avoid that construction in asserting its patent against Sun. *Southwall Techs., Inc., v. Cardinal IG Co.*, 54 F.3d 1570, 1576 (Fed. Cir. 1995) ("Claims may not be construed one way in order to obtain their allowance and in a different way against accused infringers."); *Rheox, Inc. v. Entact, Inc.*, 276 F.3d 1319, 1325 (Fed. Cir. 2002) ("Explicit arguments made during prosecution to overcome prior art can lead to a narrow claim interpretation because '[t]he public has a right to rely on such definitive statements made during prosecution.'") (quoting *Digital Biometrics, Inc. v. Identix, Inc.*, 149 F.3d 1335, 1347 (Fed. Cir. 1998)); s*ee also Koepnick Med. & Educ. Res. Found., LLC v. Alcon Labs., Inc.*, 347 F. Supp. 2d 731, 742 (D. Ariz. 2004) (holding that failure to rebut examiner's reason for allowance may be used to determine the meaning of claim limitations).

Accordingly, if the Court determines these claims are not indefinite, these claim phrases should be construed in the same manner NetApp construed them during prosecution (and as Sun now proposes), *i.e.*, to mean "associating each data block with a respective one of the storage blocks across the plurality of stripes as an association."

(ii) No Basis Exists For NetApp's Elimination Of The Express "Storage Blocks" Requirement In Favor Of Its "Locations" Construction.

NetApp's proposed construction replaces the claim term "storage blocks" with the more general term "locations." As demonstrated above, both the specification and the prosecution history clearly require storage blocks. Accordingly, NetApp's attempt to rewrite the claim

language should be rejected.

VI. CONCLUSION

For the above reasons, Sun requests the Court adopt its proposed constructions.

Dated: July 7, 2008

DLA PIPER US LLP

By /s/ Mark Fowler
Mark D. Fowler

Attorneys for Defendant and Counterclaim Plaintiff
SUN MICROSYSTEMS, INC.